dotnet new webapi -n Restaurants.API --no-openapi -controllers

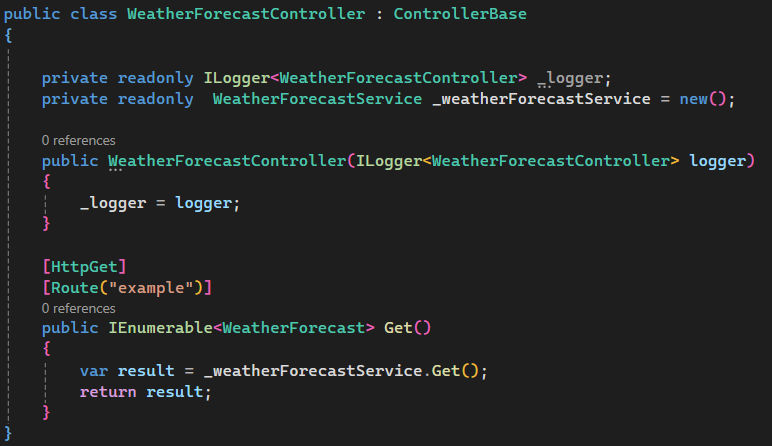
dotnet new sln

dotnet new sln -n Restaurants ---> to custom name

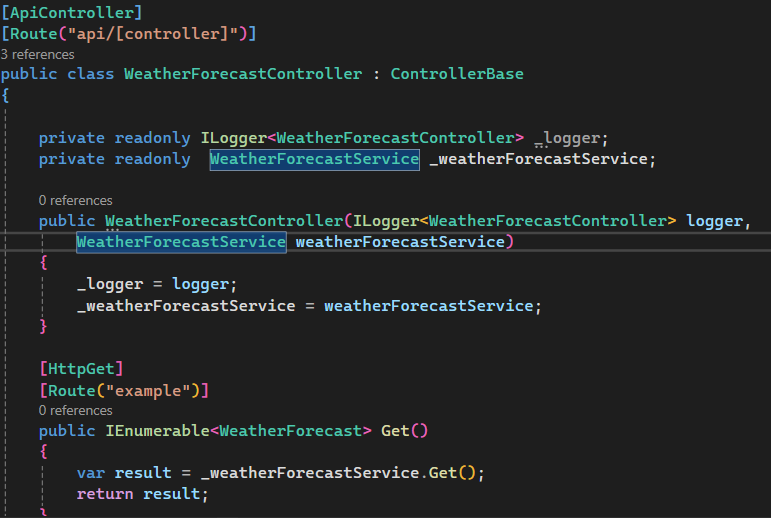
dotnet sln add ./Restaurants.API--->connecting project to solution file

<https://www.youtube.com/watch?v=E6sUJWwZLwE>

Creating and using Services

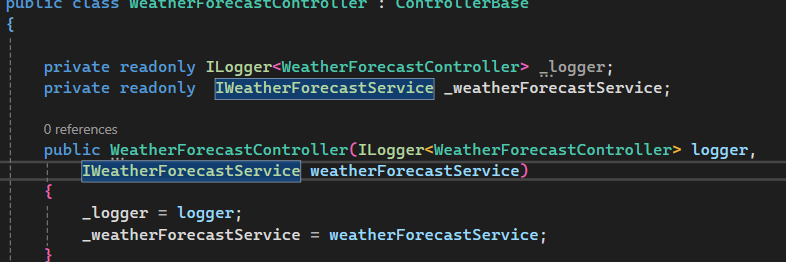


Or



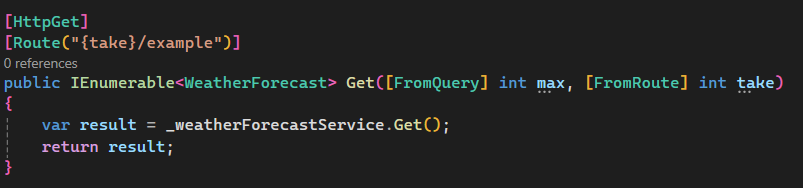
And add in program.cs 🡺 builder.Services.AddScoped<WeatherForecastService>();



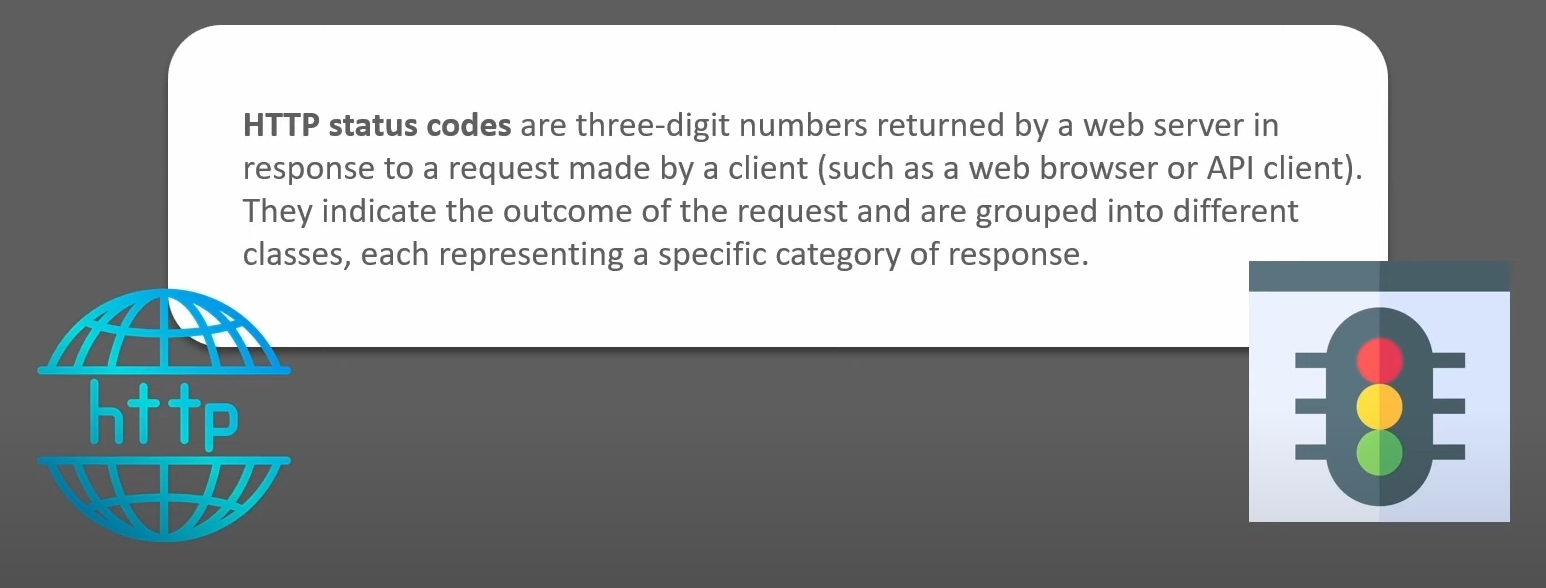
Same Service class but using interface  
  
  


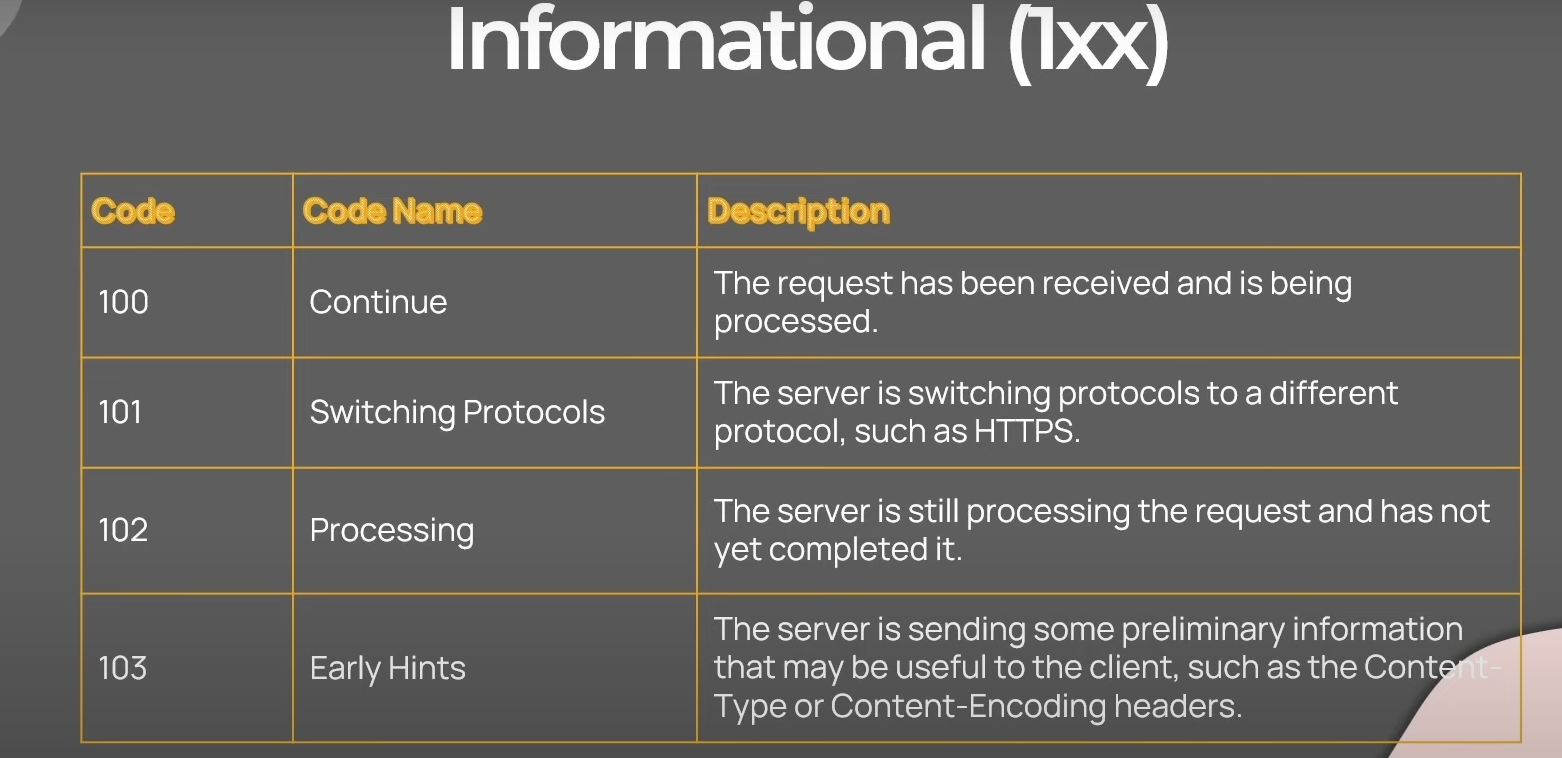
Program.cs  

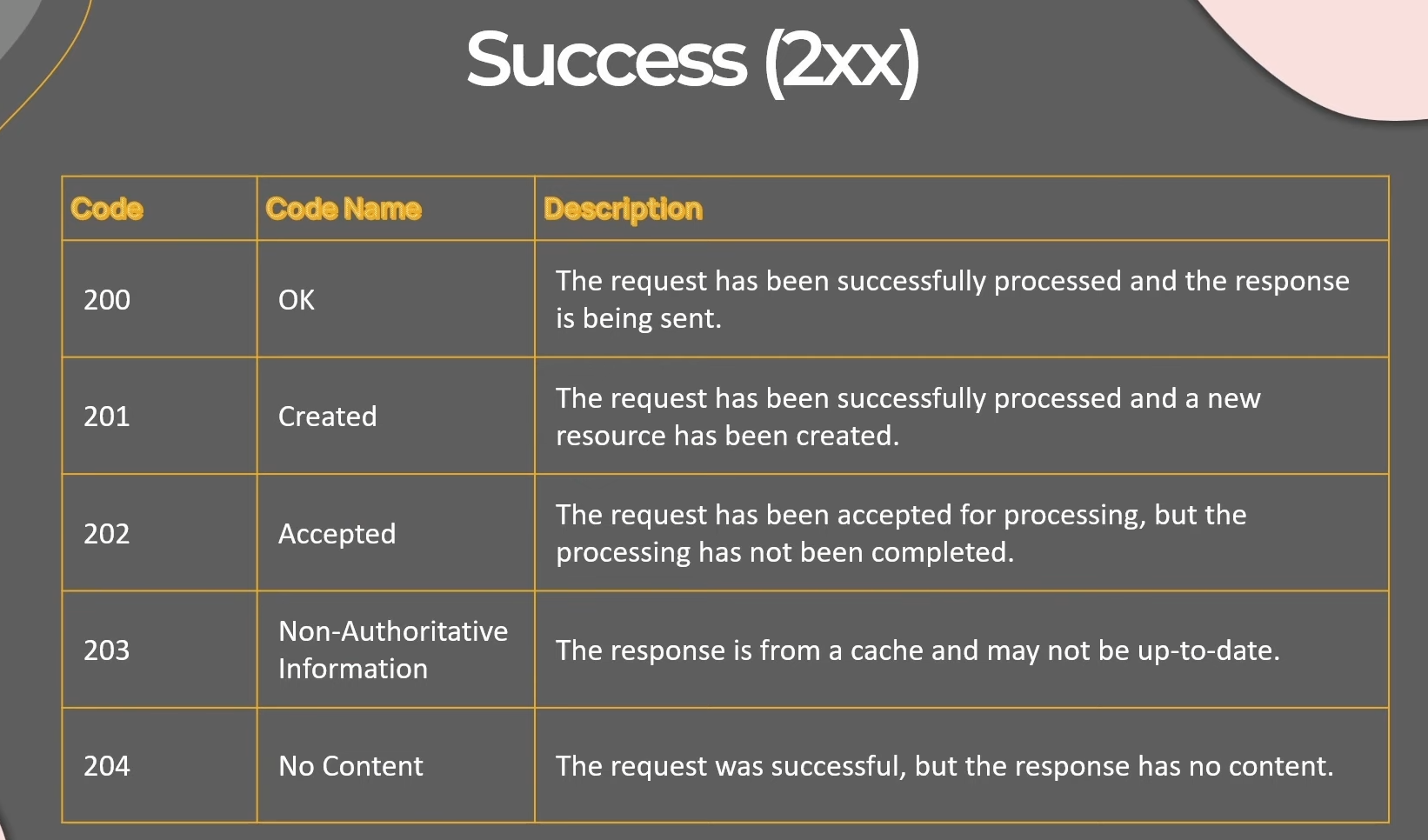

Adding Route and showing different Request Type (query, path and body)

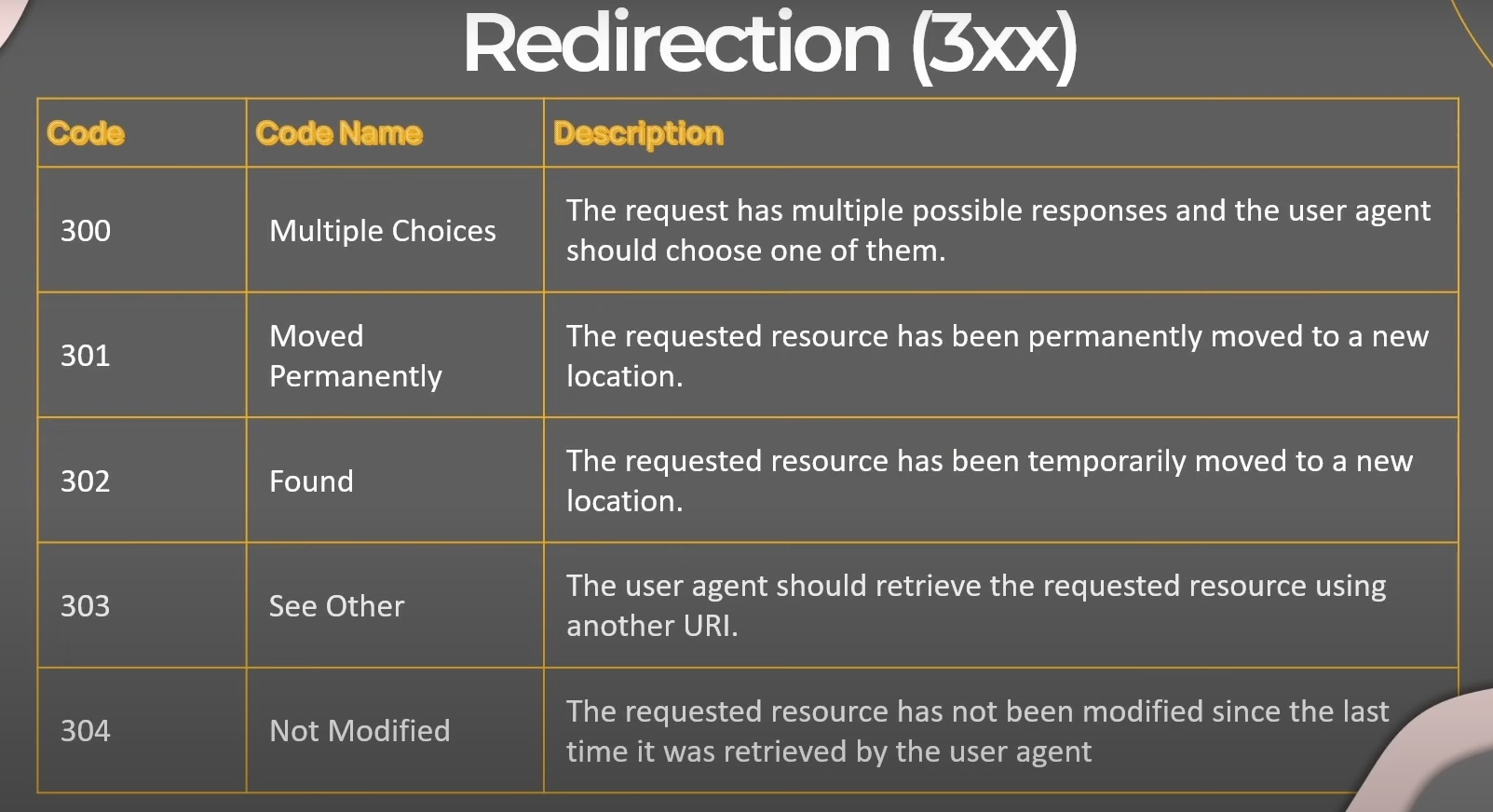


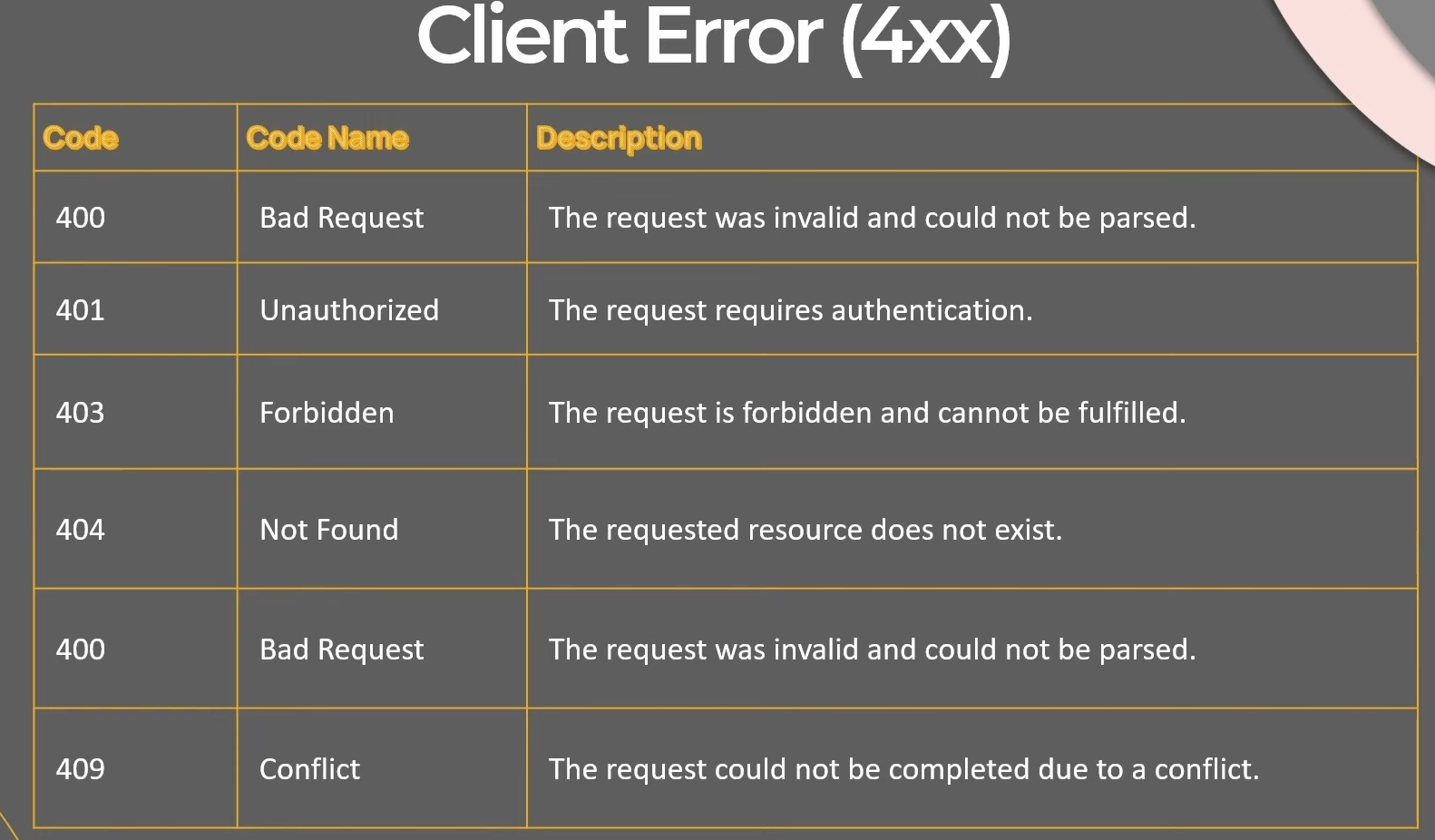


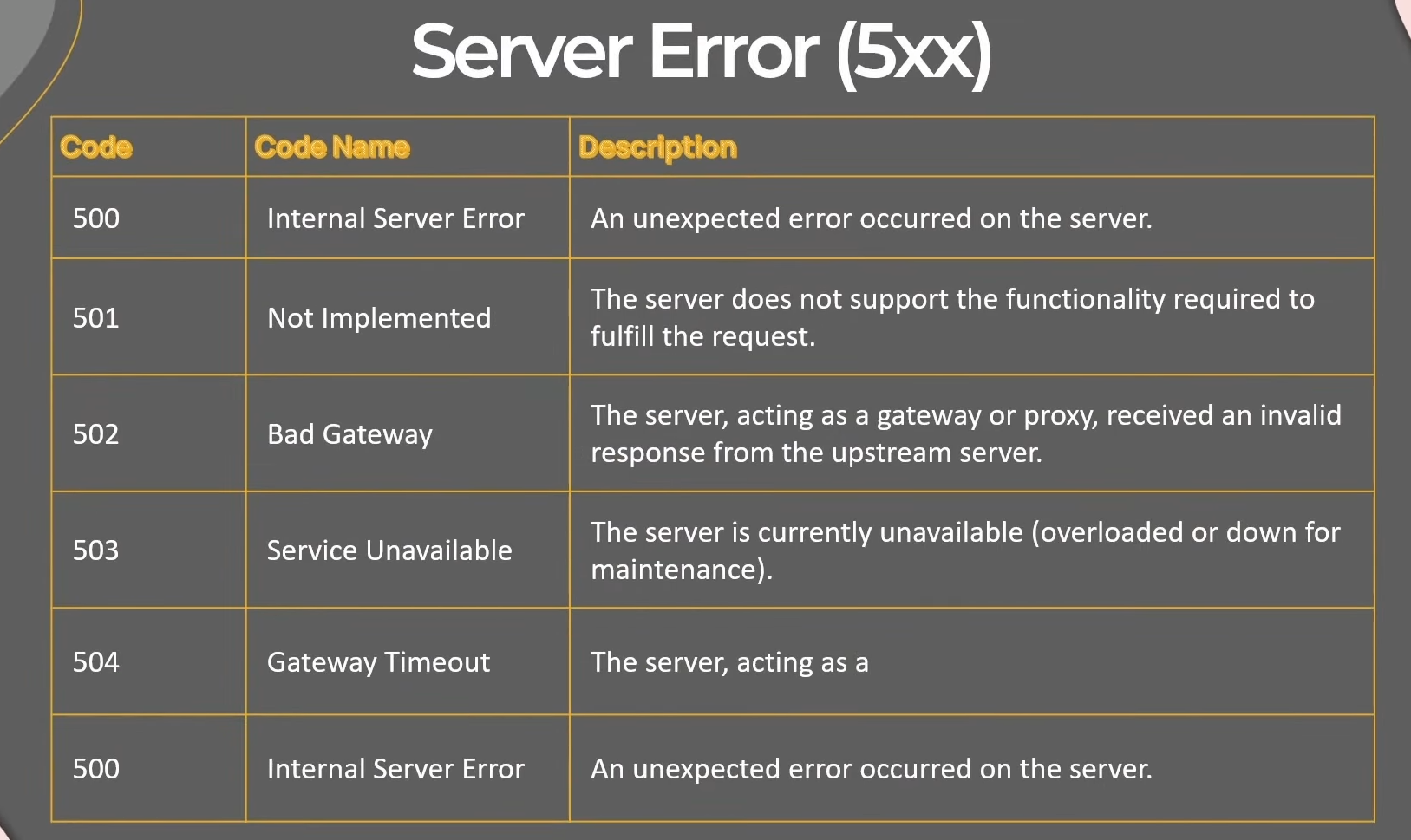


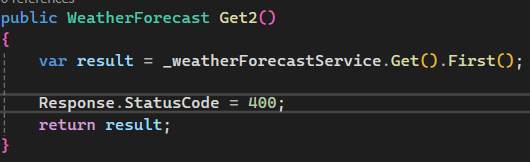
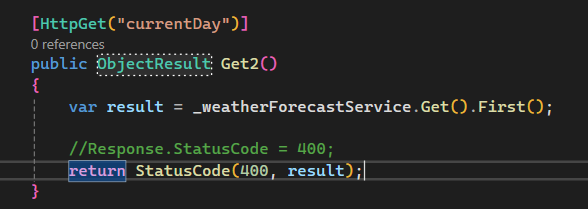




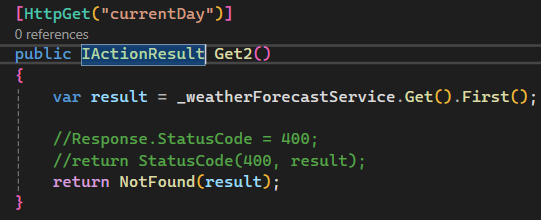


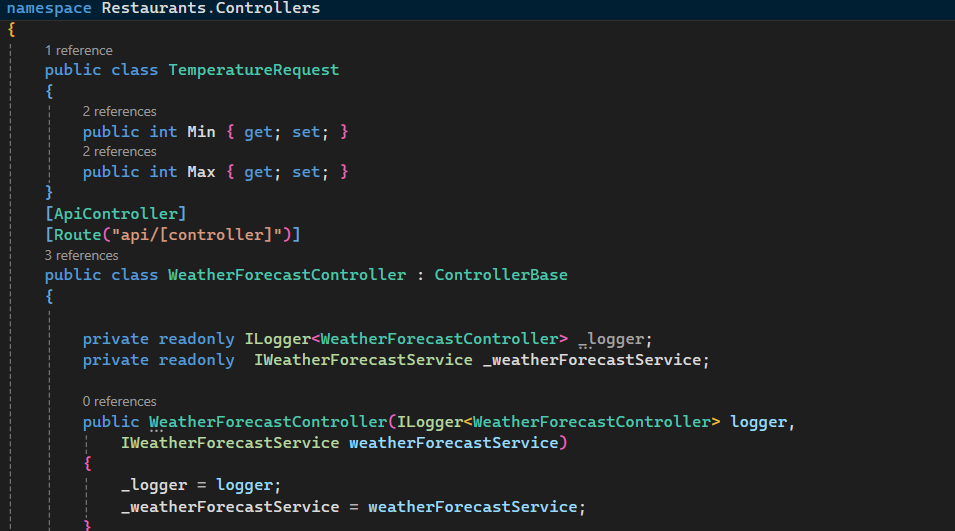




Changing Status Code in controller Response  
 or 

return BadRequest(result); return Ok(result); return Created(result); return NotFound(result);



Sample Controller  


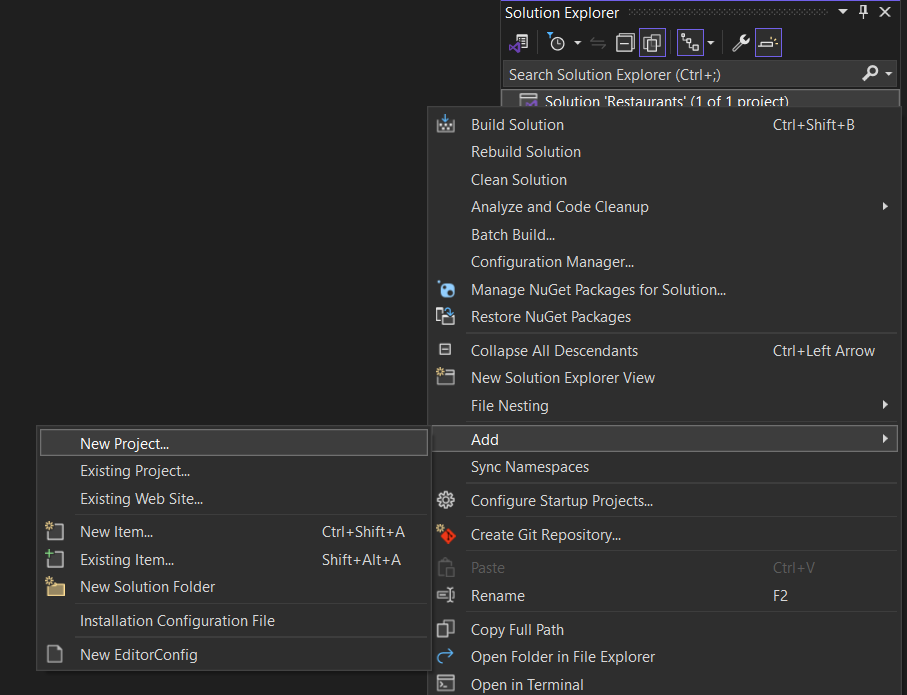


Clean Architecture  
-Domain

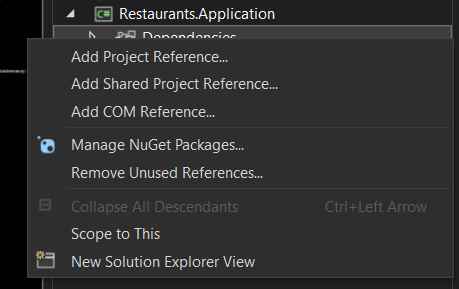
-Application

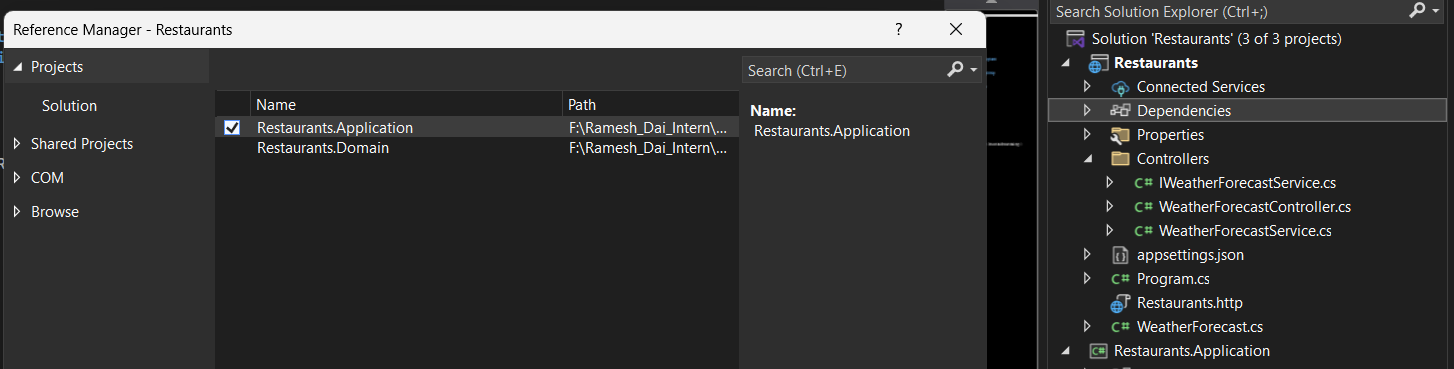
-Presentation and Infrastructure

Add Domain Project to Solution

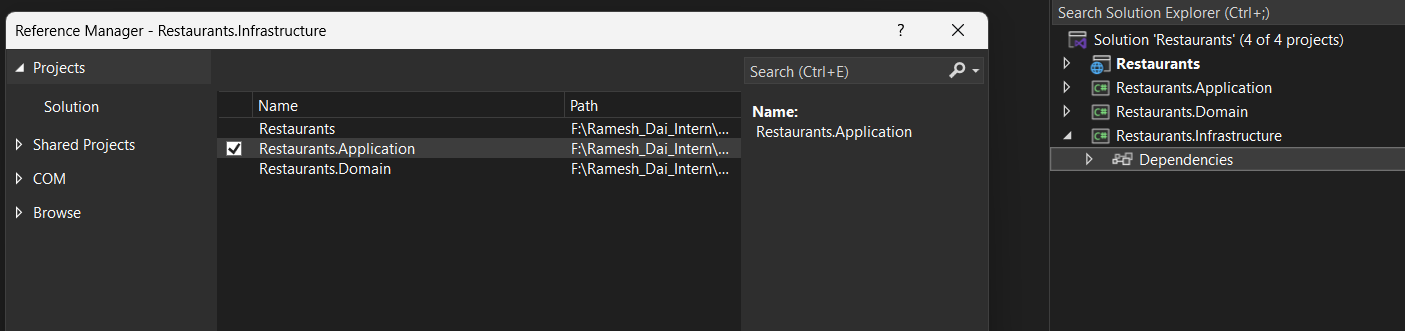
  
  
Creater Restaurants.Domain as Class Library

Create Restaurants.Application as Class Library

* Add reference of Domain for Application Dependancy
  + 
* Add reference of Application for Main Restaurants API dependencies



* Add reference of Application for Infrastructure dependencies

  
  
Shift + F2 to create new Folder and new Class

If want to create folder add slash at the end without extension cs

Creating First Entities

namespace Restaurants.Domain.Entities

{

public class Restaurant

{

public int Id { get; set; }

public string Name { get; set; } = default!;

public string Description { get; set; } = default!;

public string Category { get; set; } = default!;

public bool HasDelivery { get; set; }

public string? ContactEmail { get; set; }

public string? ContactNumber { get; set; }

public Address? Address { get; set; }

public List<Dish> Dishes { get; set; } = new();

}

}namespace Restaurants.Domain.Entities

{

public class Address

{

public string? City { get; set; }

public string? Street { get; set; }

public string? PostalCode { get; set; }

}

}namespace Restaurants.Domain.Entities

{

public class Dish

{

public int Id { get; set; }

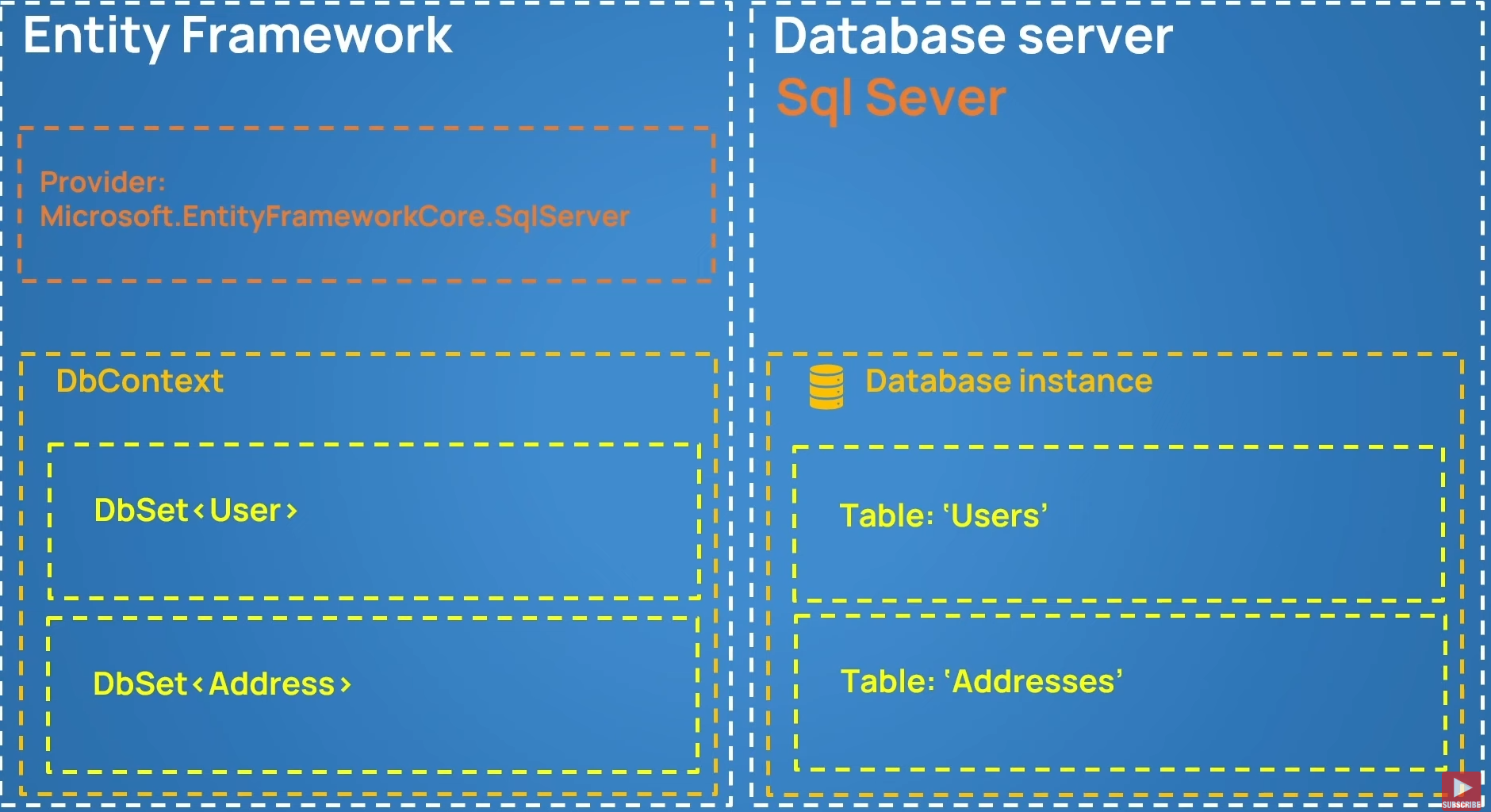
public string? Name { get; set; } = default!;

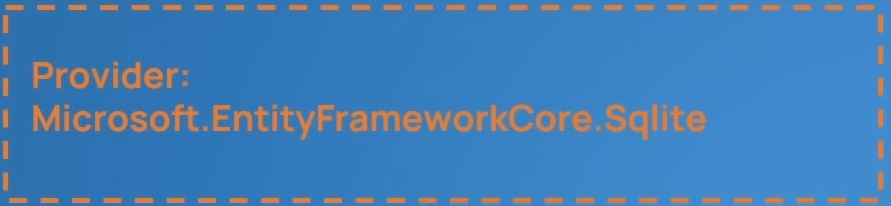
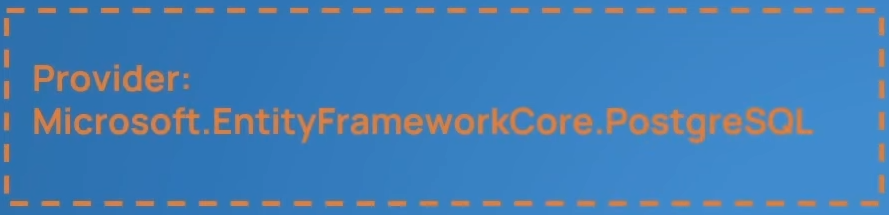
public string? Description { get; set; } = default!;

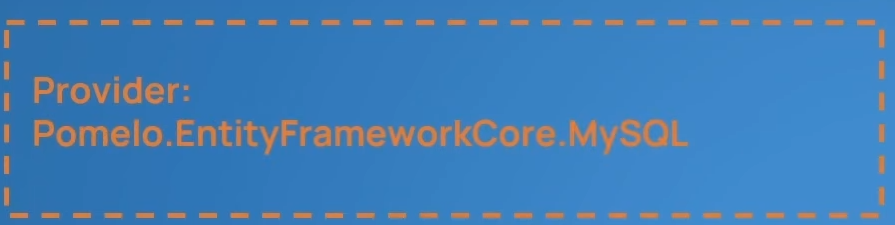
public decimal Price { get; set; }

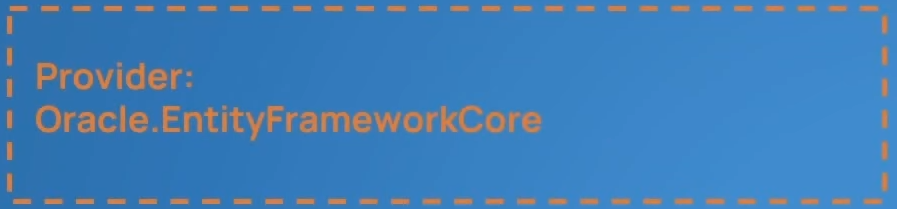
}

}







Creating Dbcontext and Registering Tables  
using Microsoft.EntityFrameworkCore;

using Restaurants.Domain.Entities;

namespace Restaurants.Infrastructure.Persistence

{

internal class RestaurantsDbContext : DbContext

{

internal DbSet<Restaurant> Restaurants { get; set; }

internal DbSet<Dish> Dishes { get; set; }

protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)

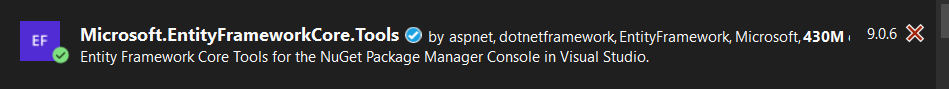
{

optionsBuilder.UseSqlServer("Server=DESKTOP-DMBAGSB\\SQLEXPRESS;Database=RestaurantsDb;Trusted\_Connection=True");

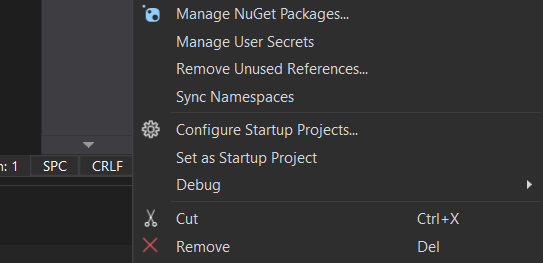
}

}

}



Add nugget package in infrastrucuture to perform migrations

Set Restaurants.Infrastructure as startup project  
  
go to package manager console

And type add-migration intit

Adding Addresstable inside the restaurant

protected override void OnModelCreating(ModelBuilder modelBuilder)

{

base.OnModelCreating(modelBuilder);

modelBuilder.Entity<Restaurant>()

.OwnsOne(r => r.Address);

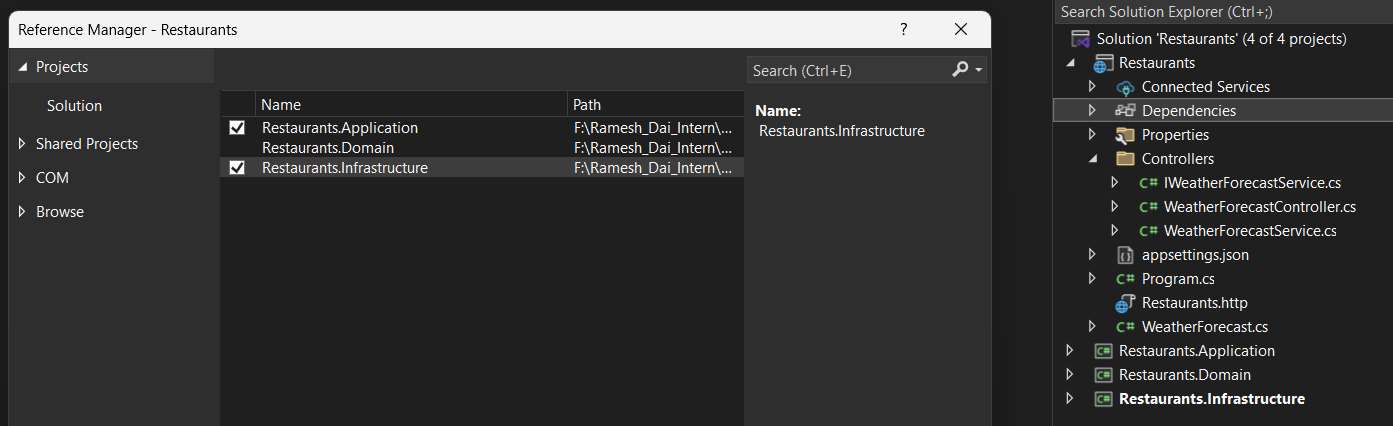
modelBuilder.Entity<Restaurant>()

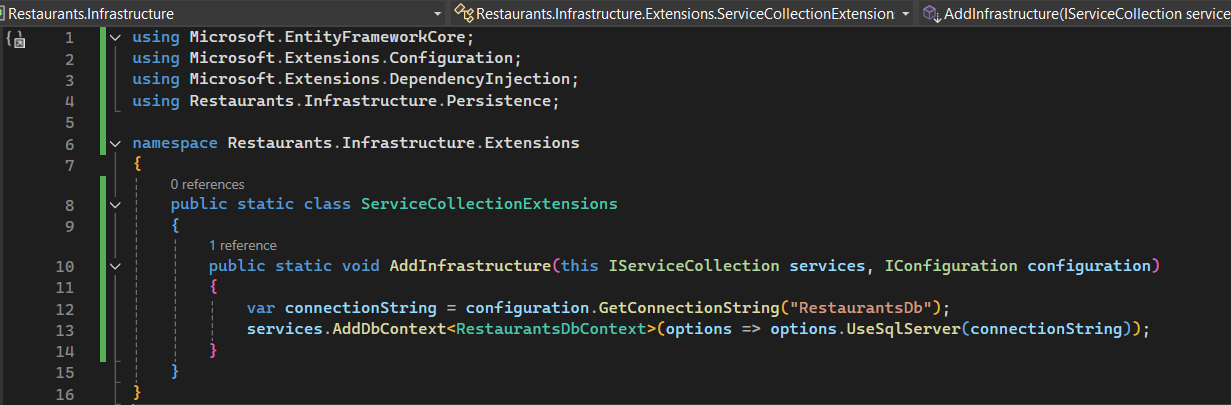
.HasMany(r => r.Dishes)

.WithOne()

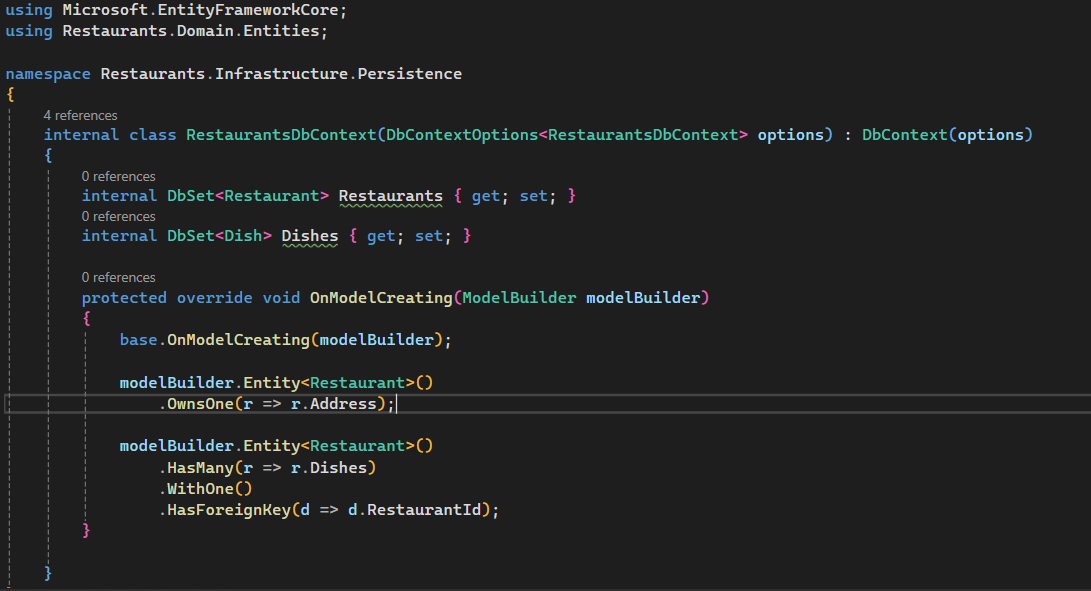
.HasForeignKey(d => d.RestaurantId);

}

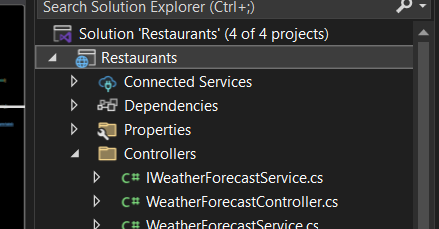
Connecting Infrastructure layer to main presentation layer  
For connecting database   


Restaurants.Infrastructure.Extensions/ServiceCollectionExtensions.cs  


RestaurantDbContext.cs



set Restaurants.API as startupProject.

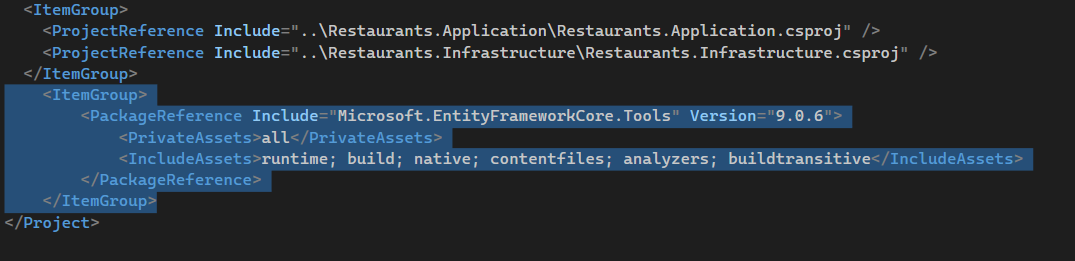
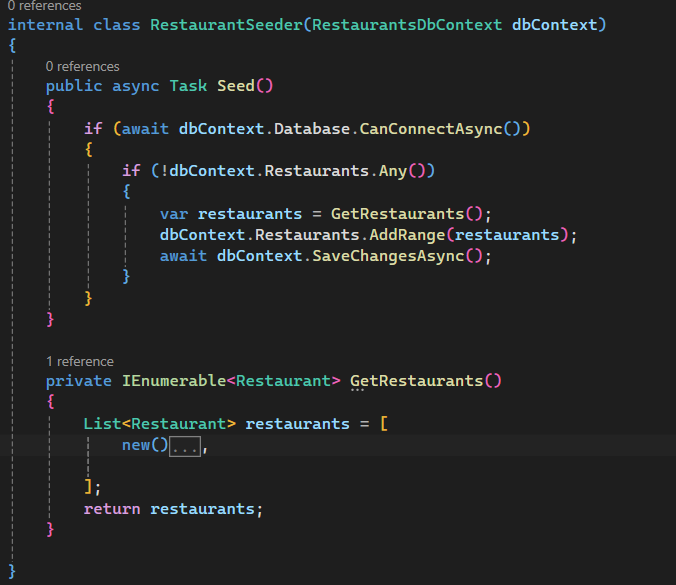
  
  
 <ItemGroup>

<PackageReference Include="Microsoft.EntityFrameworkCore.Tools" Version="9.0.6">

<PrivateAssets>all</PrivateAssets>

<IncludeAssets>runtime; build; native; contentfiles; analyzers; buildtransitive</IncludeAssets>

</PackageReference>

</ItemGroup>  
  
Add this to Restaurants to API to be able to update database when restaurants.API is setup as startup Project  
  
  
Seeding the Database  


List<Restaurant> restaurants = [

new()

{

Name = "KFC",

Category = "Fast Food",

Description = "KFC is an AMerican food Company",

ContactEmail = "caont@yahoo.com",

HasDelivery = true,

Dishes =

[

new()

{

Name = "Nashville Hot Chicken",

Description = "Nash (10pcs)",

Price = 10.30M,

},

new()

{

Name = "Chicken Nuggets",

Description = "Nuggets (5pcs)",

Price = 5.30M,

},

],

Address = new()

{

City = "London",

Street = "Cork St 5",

PostalCode = "WC2N 5DU",

}

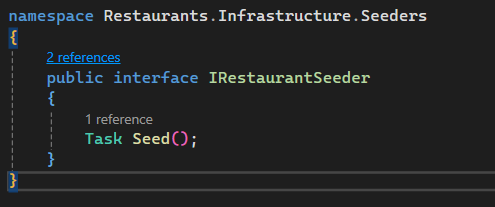
},

];

Also change this



Change this



Program.cs

